

In the Claims:

Please cancel claims 1-3, 7-8, 17-18, and 21-23, replace claims 4-6, 9-10, 13, 16, and 24-25, and add new claim 26, all as shown below.

1. - 3. (Canceled)

4. (Currently Amended): The computer-readable ~~machine-readable~~ medium of claim 26 wherein the superclass includes logic to handle server side tasks.

5. (Currently Amended): The computer-readable ~~machine-readable~~ medium of claim 26 wherein the wrapper class is generated in bytecode.

6. (Currently Amended): The computer-readable ~~machine-readable~~ medium of claim 5 wherein bytecode is generated for vendor methods not implemented in the superclass.

7. - 8. (Canceled):

9. (Currently Amended): The computer readable medium ~~machine-readable medium~~ of claim 8 26, wherein the ~~standard features are J2EE~~ application server supports Java Enterprise Edition features.

10. (Currently Amended): A computer-readable machine-readable medium carrying one or more sequences of instructions for processing an invocation at a dynamically generated wrapper, which instructions, when executed by one or more processors, cause the one or more processors to carry out comprising the steps of:

receiving, from an application program, an invocation call directed to a wrapped vendor object resource adapter;

initiating pre-processing by calling a pre-invocation handler configured to execute server-side code;

calling the wrapped vendor object;

receiving a result from the wrapped vendor object;

initiating post-processing by calling a post-invocation handler configured to execute post processing server-side tasks; and

providing the result to the application program, thereby enabling the application program to access vendor specific extension methods of the wrapped resource adapter vendor object.

11. – 12. (Canceled)

13. (Currently Amended): The computer-readable machine-readable medium of claim 10 wherein the server-side code executed by the pre-invocation handler includes global transaction processing code.

14. – 15. (Canceled)

16. (Currently Amended): The computer-readable machine-readable medium of claim 10 wherein the post-processing server-side tasks include ~~global~~ transaction management.

17. – 23. (Canceled)

24. (Currently Amended): A computer-readable machine-readable medium carrying ~~one or more~~ sequences of instructions for processing an invocation at a dynamically generated wrapper, ~~which~~ instructions, when executed by ~~one or more~~ processors, ~~cause the one or more processors to carry out~~ comprising the steps of:

receiving, from an application ~~program~~, a method invocation ~~call~~ directed to a resource adapter ~~vendor object~~;

calling a wrapper object for processing the method invocation ~~call~~ wherein the wrapper object ~~has been~~ is dynamically generated from a resource adapter class ~~vendor class~~ to be associated with the ~~vendor object~~;

initiating pre-processing by the wrapper object, wherein the wrapper object calls a pre-invocation handler configured to perform server side logic;

forwarding the method invocation ~~call~~ to the resource adapter ~~vendor object~~ by the wrapper object on behalf of the application ~~program~~;

receiving a result of the method invocation ~~call~~ from the resource adapter ~~vendor object~~ by the wrapper object;

initiating post-processing by the wrapper object, wherein the wrapper object calls a post-

invocation handler configured to perform server-side logic; and

providing the result to the application ~~program~~, thereby enabling the application ~~program~~ to access vendor specific extension methods of the resource adapter ~~vendor object~~.

25. (Currently Amended): The computer-readable ~~machine-readable~~ medium of claim 24 wherein the server-side logic includes at least one of ~~global~~ transaction management, pooling, caching, tracing and profiling.

26. (New): A computer-readable medium carrying instructions for dynamically generating a wrapper object, comprising the steps of:

receiving a resource adapter class at an application server;

performing reflection on the resource adapter class to identify interfaces implemented by the resource adapter class;

dynamically generating a wrapper class at runtime that extends from a superclass, wherein the superclass implements Java Database Connectivity, Java Messaging Service, or Java Connector Architecture interfaces, and the wrapper class implements the interfaces identified through reflection;

instantiating a wrapper object from the wrapper class; and

providing the wrapper object to an application that requires support for the interfaces implemented by the resource adapter class.